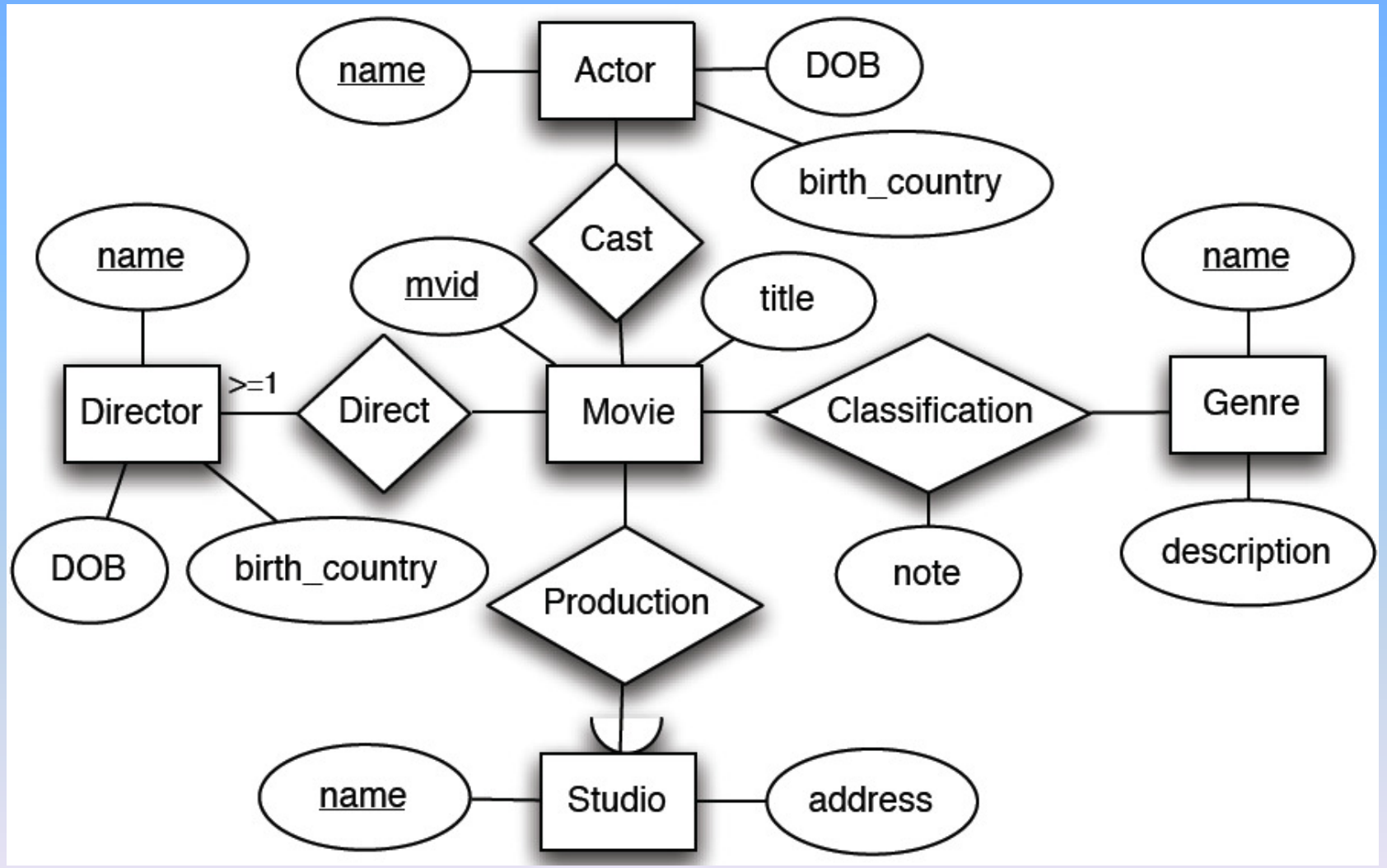
The Entity-Relationship Model 2

- Advanced concepts for ER Modelling
  - –Multiway relationships
  - –Sub-classes
  - –Weak entity set
- •Design techniques
- •Map ER models into relational database schemas

Reading: Sections 4.1.7, 4.2, 4.4, 4.5.3, 4.5.4 of Textbook.

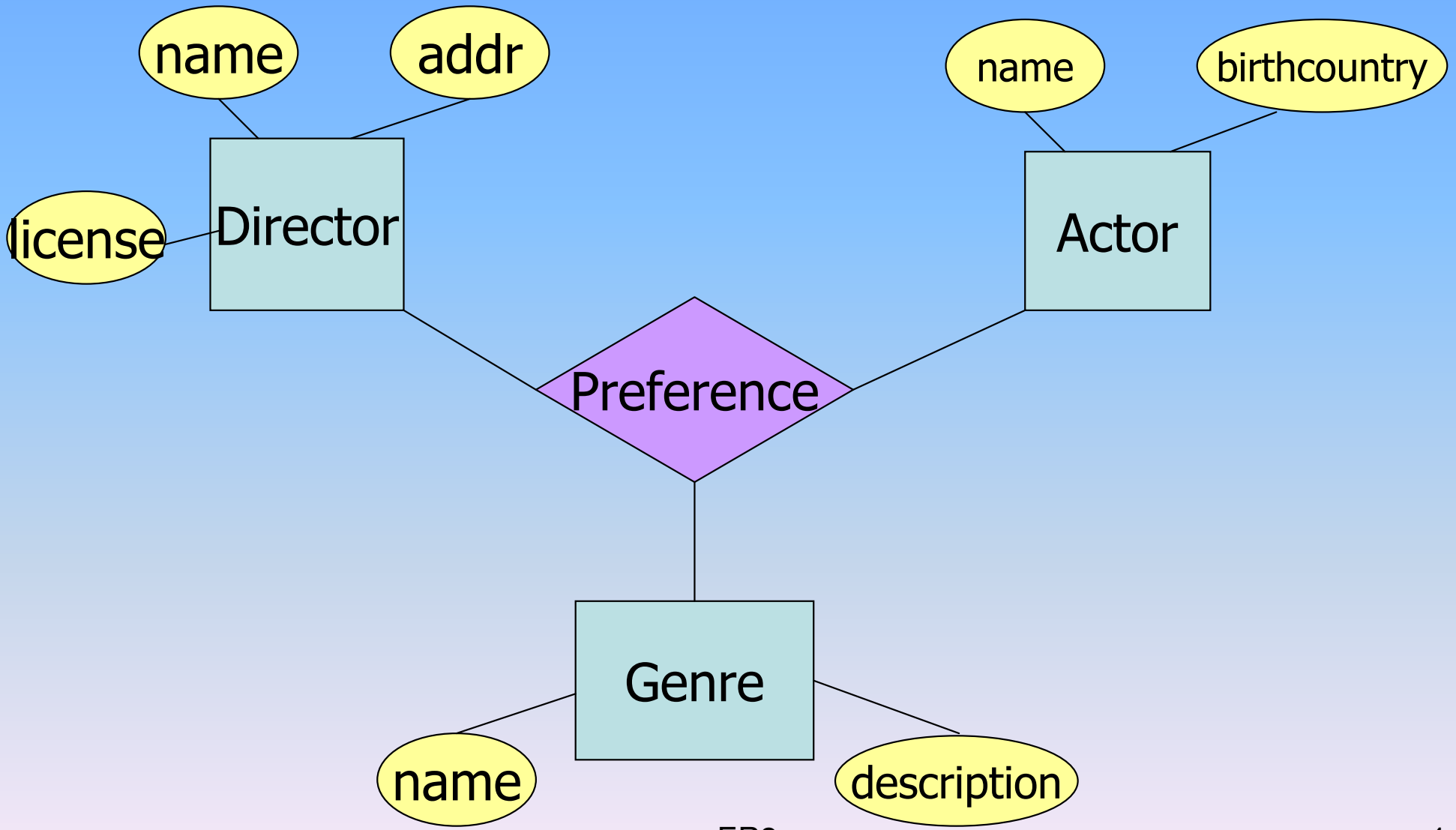
The E/R model for the Village Cinema database



Multiway Relationships

- Multiway relationships connect more than two entity sets.
- Suppose that some directors prefer to collaborate with some actors for some genres of movies.
  - This is a 3-way relationship among Directors, Actors and Genres.
  - Our three binary relationships Direct, Cast, and Classification are not about this relationship.

Example: 3-Way Relationship



Weak Entity Sets

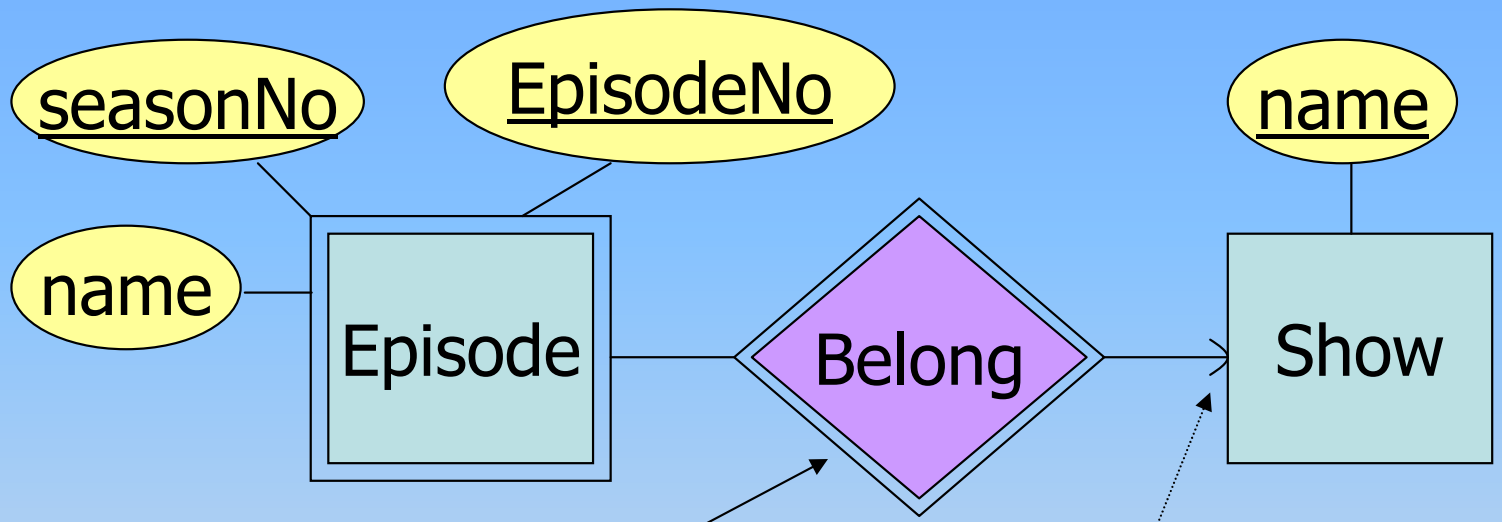
- Occasionally, entities of an entity set need "help" to identify them uniquely.
- Entity set *E* is said to be weak if in order to identify entities of *E* uniquely, we need to follow one or more many-one relationships from *E* and include the key of the related entities from the connected entity sets.

Weak Entity Set: Example

A TV-show like The Simpsons, South Park, or Neighbours has many episodes.

- Some show episodes even do not have a title, and so title can not, and usually is not used to uniquely identify a show episode.
- Season number and Episode number are certainly not a key, since two episodes from different shows can have the same Season and Episode numbers.
- But season and number, together with the show name where the episode belongs to should be unique.

In E/R Diagrams



Note: This is NOT an ordinary relationship between entities.

Note: must be rounded because each player needs a team to help with the key.

- Double diamond for *supporting* many-one relationship.
- Double rectangle for the weak entity set.

Weak Entity-Set Rules

- A weak entity set has one or more many-one relationships to other (supporting) entity sets.
  - Not every many-one relationship from a weak entity set need be supporting.
  - But supporting relationships must have a rounded arrow (entity at the “one” end is guaranteed).

Weak Entity-Set Rules …

- The key for a weak entity set is its own underlined attributes and the keys for the supporting entity sets.
  - E.g., season-number and episode-number, and (show) name is a key for episodes.

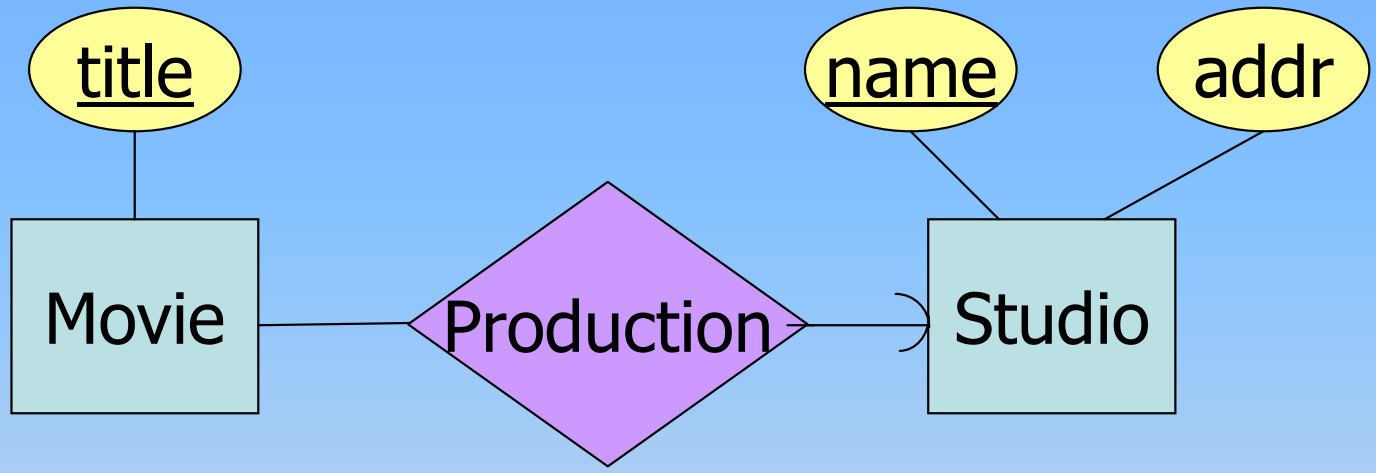
Design Techniques

1. Avoid redundancy.
2. Limit the use of weak entity sets.
3. Don’t use an entity set when an attribute will do.

Avoiding Redundancy

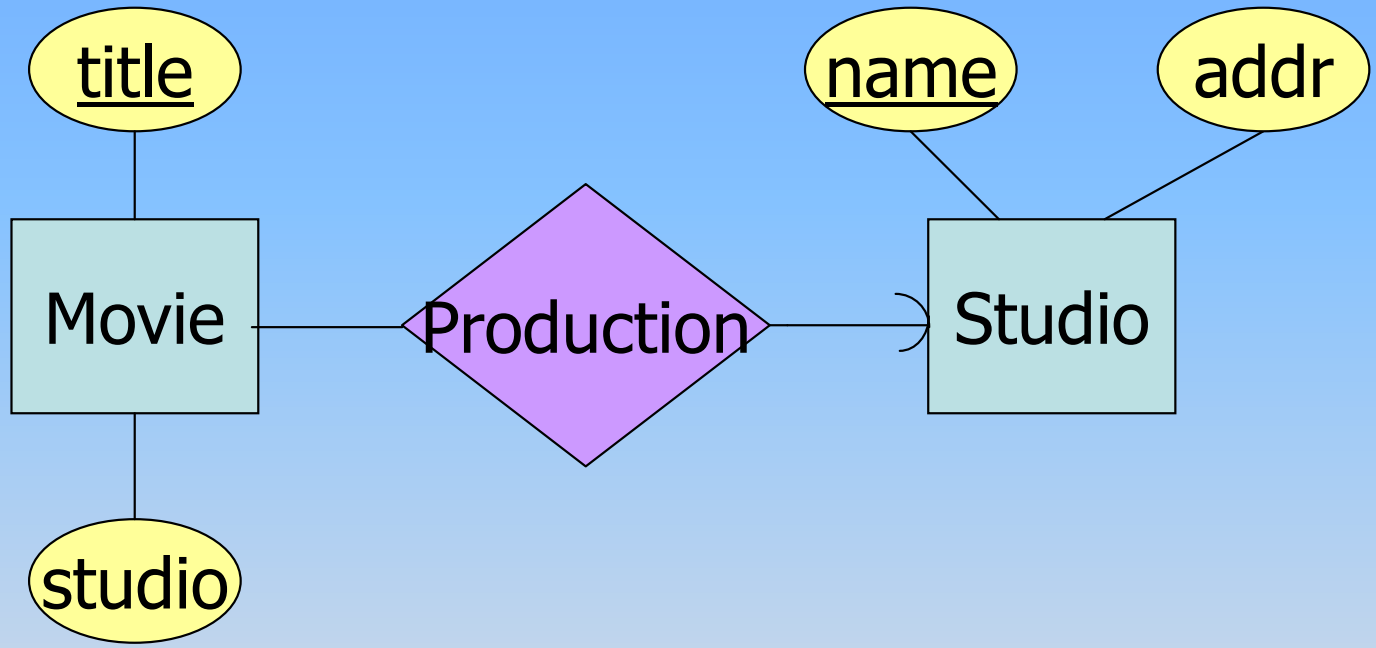
- *Redundancy:* saying the same thing in two (or more) different ways.
- Wastes space and (more importantly) encourages inconsistency.
  - Two representations of the same fact become inconsistent if we change one and forget to change the other.
  - Recall anomalies due to FD's.

Example: Good



This design gives the address of each studio exactly once.

Example: Bad



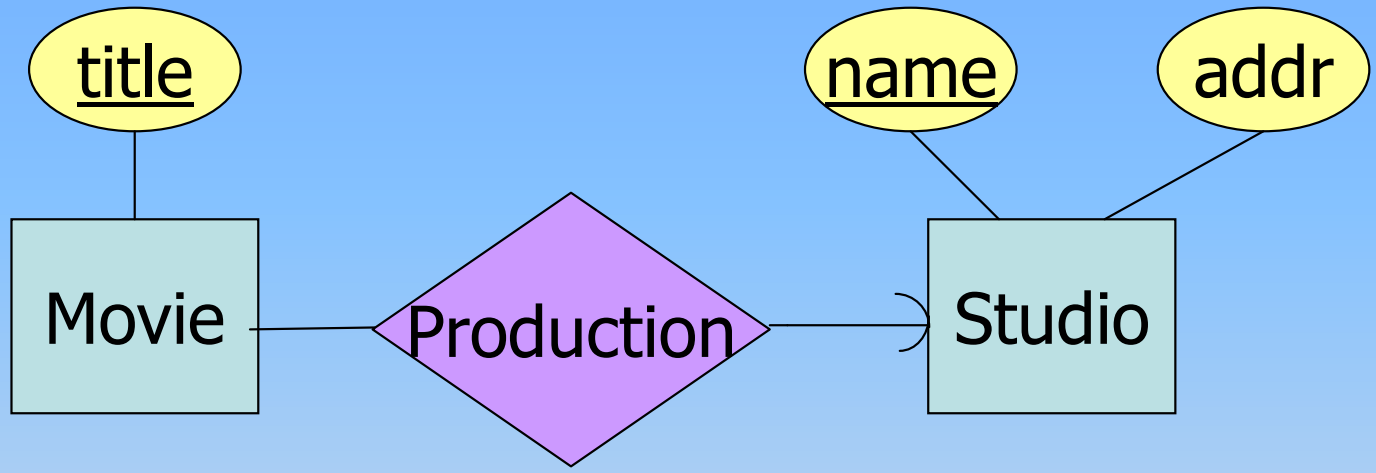
This design represents the production studio for a movie twice: as an attribute and as a related entity.

Entity Sets Versus Attributes

- An entity set should satisfy at least one of the following conditions:
  - It is more than the name of something; it has at least one nonkey attribute.

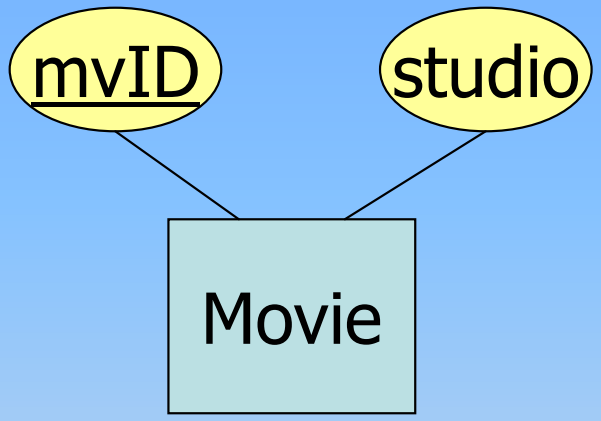
    or
  - It is the “many” in a many-one or many-many relationship.

Example: Good



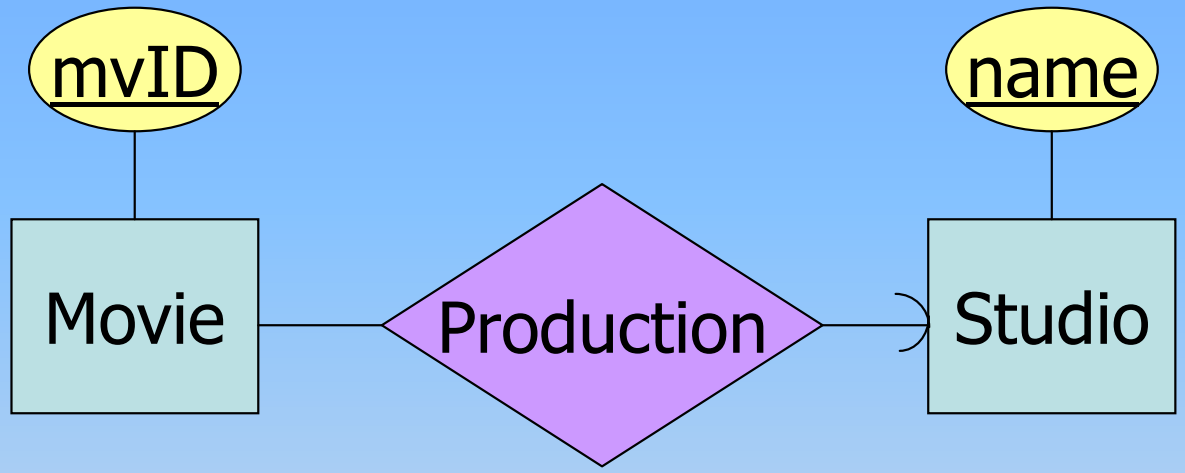
- Studio deserves to be an entity set because of the nonkey attribute addr.
- Movie deserves to be an entity set because it is the "many" of the many-one relationship Production.

Example: Good



If the details of studios such as address are not of main interest to us, there is no need to make the manufacturer an entity set. We record nothing about studios besides their name.

Example: Bad



Since Studio is nothing but a name, and is not at the "many" end of any relationship, it should not be an entity set.

Do not Overuse Weak Entity Sets

- In reality, we usually create unique ID's for entity sets.
  - Examples include social-security numbers, automobile VIN's etc.
- Sometimes even artificial ID's are used. For example mvID for Movies.
- When weak entity sets are needed?
  - The usual reason is that there is not a commonly agreed rule to create even artificial ID's for entities. For example in a TV show database, there is not a universal rule for creating ID's for episodes across all TV shows. So Episode should be a weak entity set.

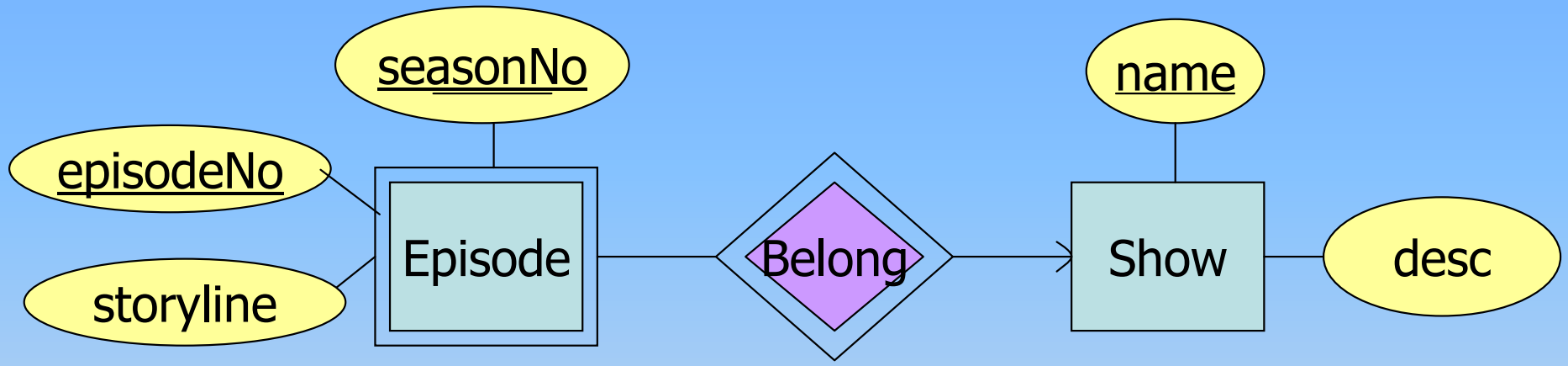
From E/R Diagrams to Relations

- Basics:
  - Entity set -> relation.
    - Attributes -> attributes.
  - Relationships -> relations whose attributes are only:
    - The keys of the connected entity sets.
    - Attributes of the relationship itself.
- Advanced:
  - Weak entity sets
  - Deriving FDs from ER diagrams
  - Simplifying relational database schema

Weak Entity Set → Relation

- Relation for a weak entity set must include attributes for its complete key (including those belonging to other entity sets), as well as its own, nonkey attributes.
- A supporting relationship is redundant and yields no relation (unless it has attributes).

Example



Show(<u>name</u>, desc)
ShowEpisode(<u>showName, seasonNo, episodeNo</u>, storyline)

Note: ShowEpisode is representing the Weak entity set Episode as well as the supporting relationship Belong.

From ER Constraints to Relational FDs

- The constraints in ER diagrams can be represented as FDs, including
  - Key of entity sets
  - The multiplicity of relationship sets
  - Weak entity sets

Note: *Most but not all* FDs are represented in ER diagram.

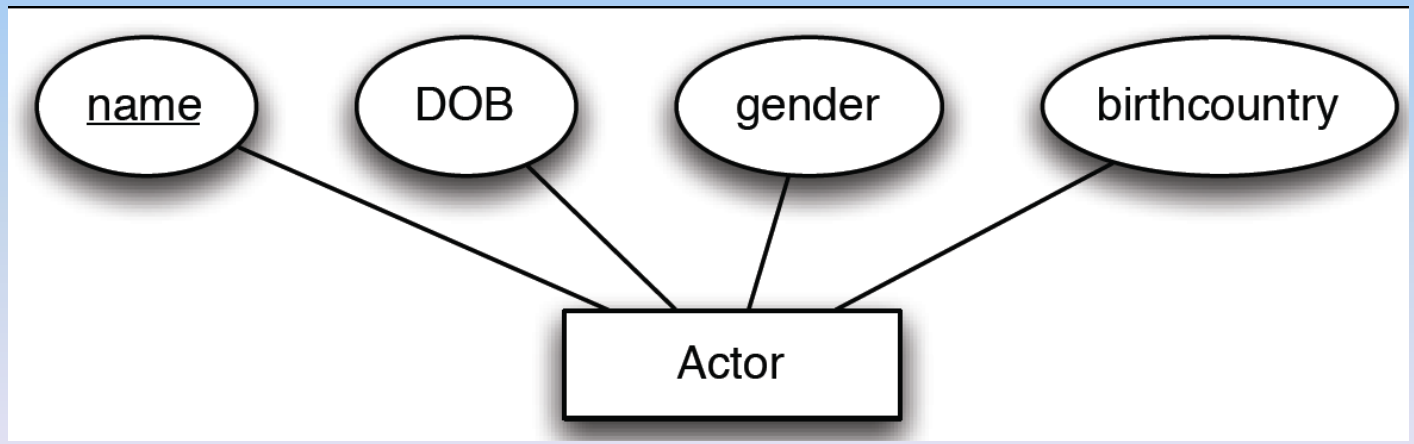
Key of Entity Sets

- The key of an entity *E* expresses the below FD (recall that *K* may comprise several attributes):

  Key→ other attributes of *R*

  For example:

  name → DOB, gender, birthcountry□

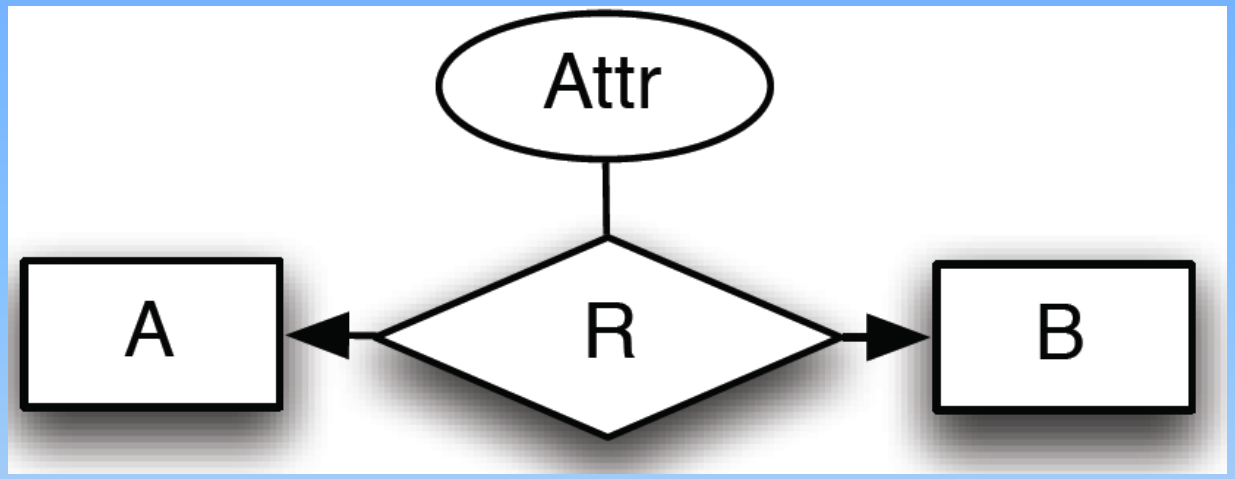


Relationships

The multiplicity of relationships expresses FDs.

- 1:1 relationship
- 1:M relationship
- M:N relationship
- Ternary relationships
  - M:N:1
  - M:1:1
  - 1:1:1
  - M:N:M

One-one Relationships

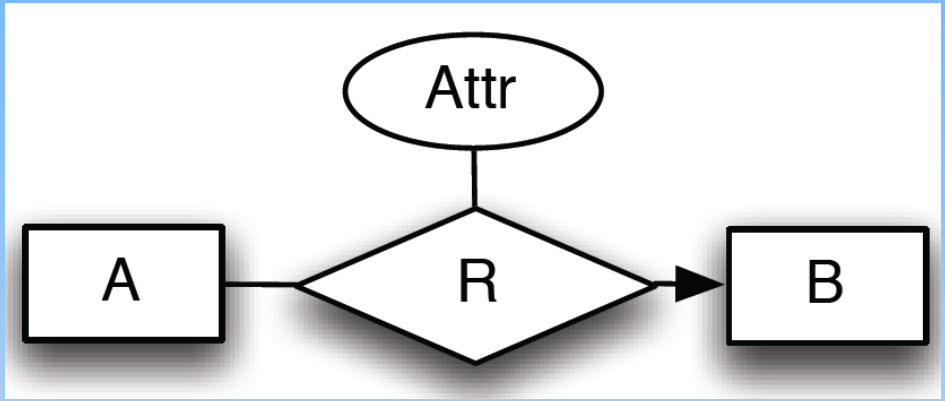


An 1:1 relationship expresses the below FDs:

Key-of-A → Key-of-B, Attr

Key-of-B → Key-of-A, Attr

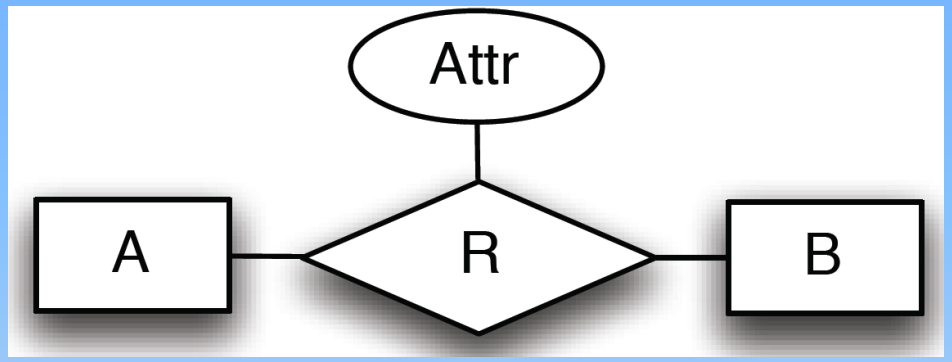
Many-one relationships



An M:1 relationship expresses the below FD:

Key-of-A → Key-of-B, Attr

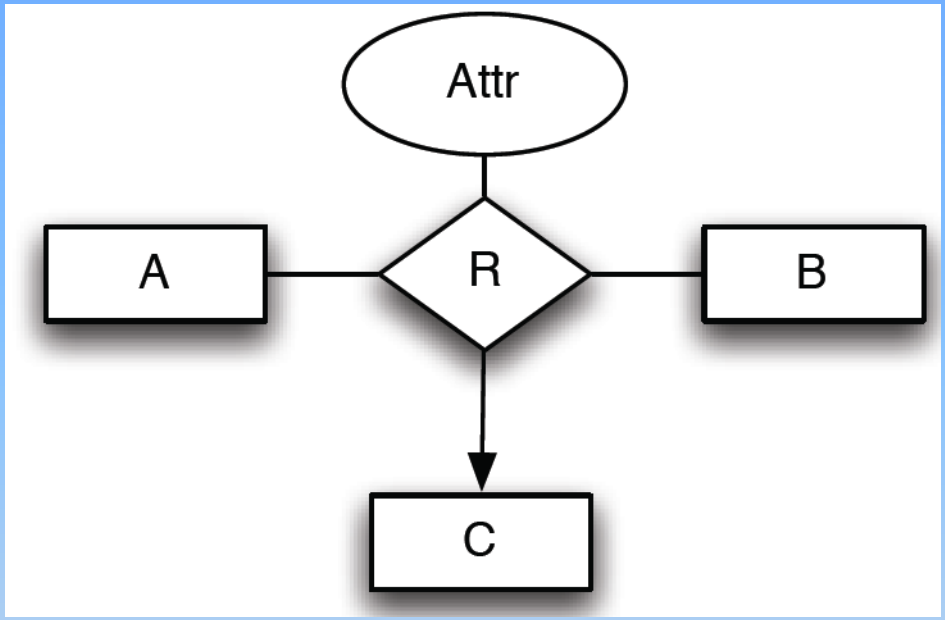
M:N Relationships



An M:N relationship expresses the below FD:

Key-of-A, Key-of-B → Attr

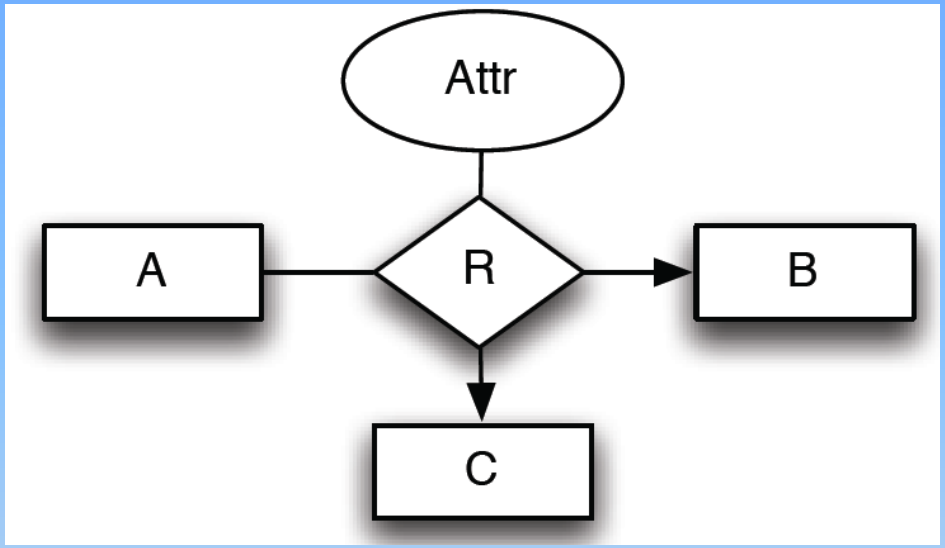
Ternary Relationships M:N:1



- An M:N:1 ternary relationship expresses a below FD:

  Key-of-A, Key-of-B → Key-of-C, Attr.

Ternary Relationships M:1:1

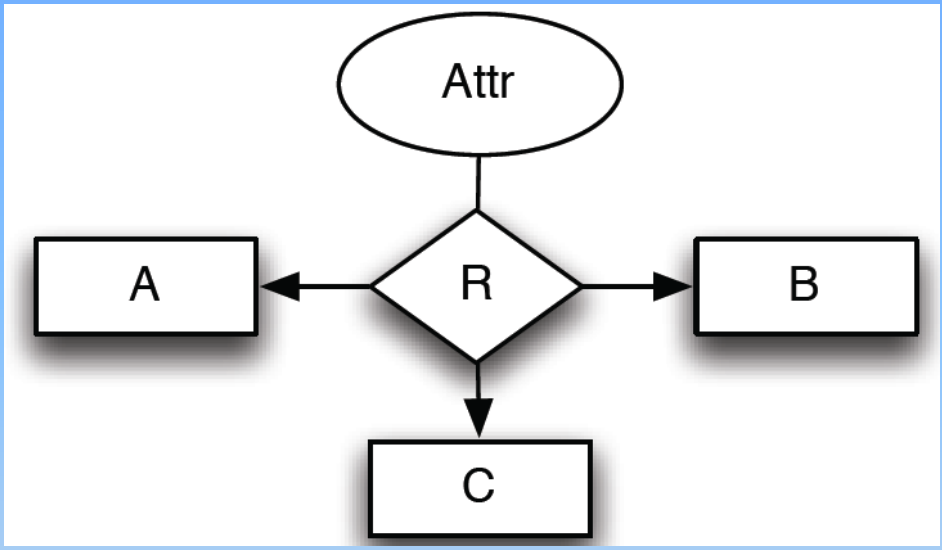


- An M:1:1 ternary relationship expresses two FDs:

  Key-of-A, Key-of-B → Key-of-C, Attr

  Key-of-A, Key-of-C → Key-of-B, Attr

Ternary Relationships 1:1:1



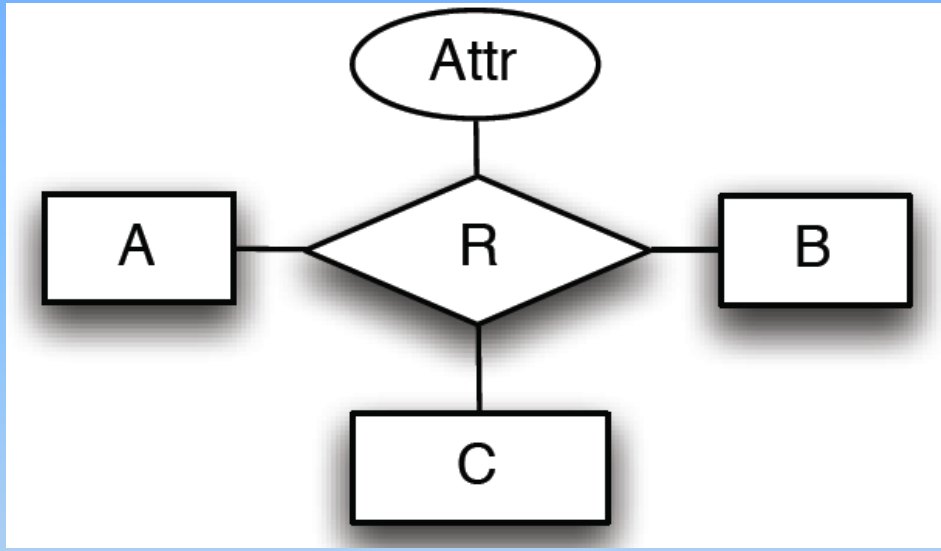
- An 1:1:1 ternary relationship expresses 3 FDs:

  Key-of-A, Key-of-B → Key-of-C, Attr

  Key-of-A, Key-of-C → Key-of-B, Attr

  Key-of-B, Key-of-C → Key-of-A, Attr

M:N:M Relationships



- An M:N:M ternary relationship expresses FD:

  Key-of-A, Key-of-B, Key-of-C → Attr

Weak entity set

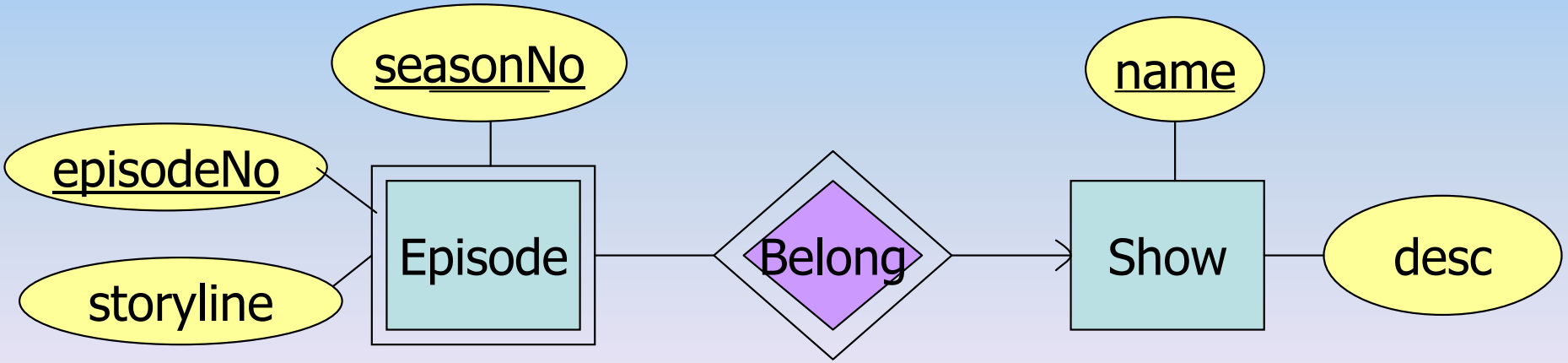
- A weak entity set supporting relationship and entity, expresses an FD:

  Key-of-supporting-entity, Key-of-weak-entity

  → *other-attributes-of-weak-entity.*

  For example,

  show_name, episodeNo, seasonNo → storyline



Merging M:1 relationships

- Relations for many-one relationships are merged with relation for the entity set on the “Many” side. The normal form of relations is kept.

Note: one-one relationship is a special many-one relationship.

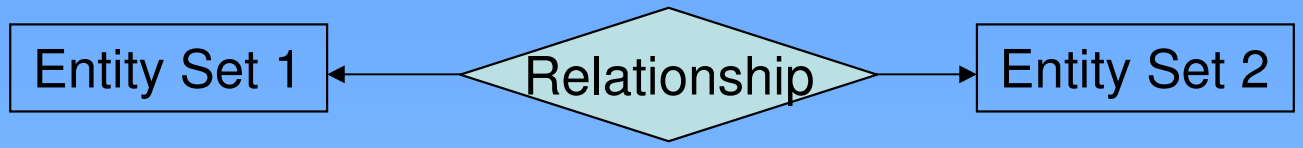
Example:

Movie(mvID, title, rating, rel_date, length)
Production(mvID, studio_name*)
Studio(name, address)

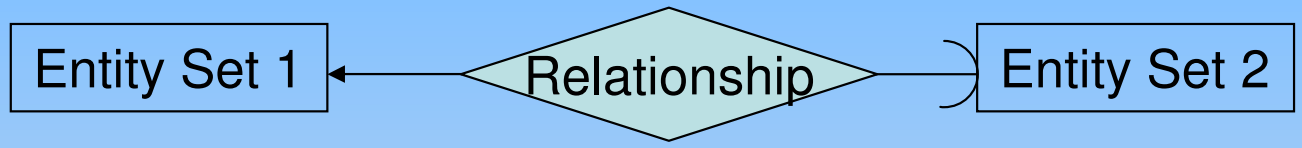
Movie and Production can be merged:

Movie(mvID, title, rating, rel_date, length, studio_name*)
Studio(name, address)

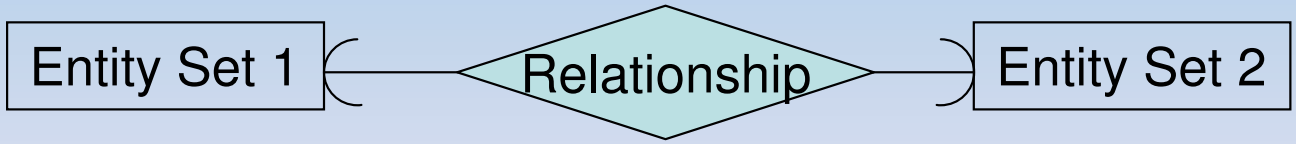
Merging 1:1 Relationships



- Merging relationship relation with either side.
  - Null represents nil relationship

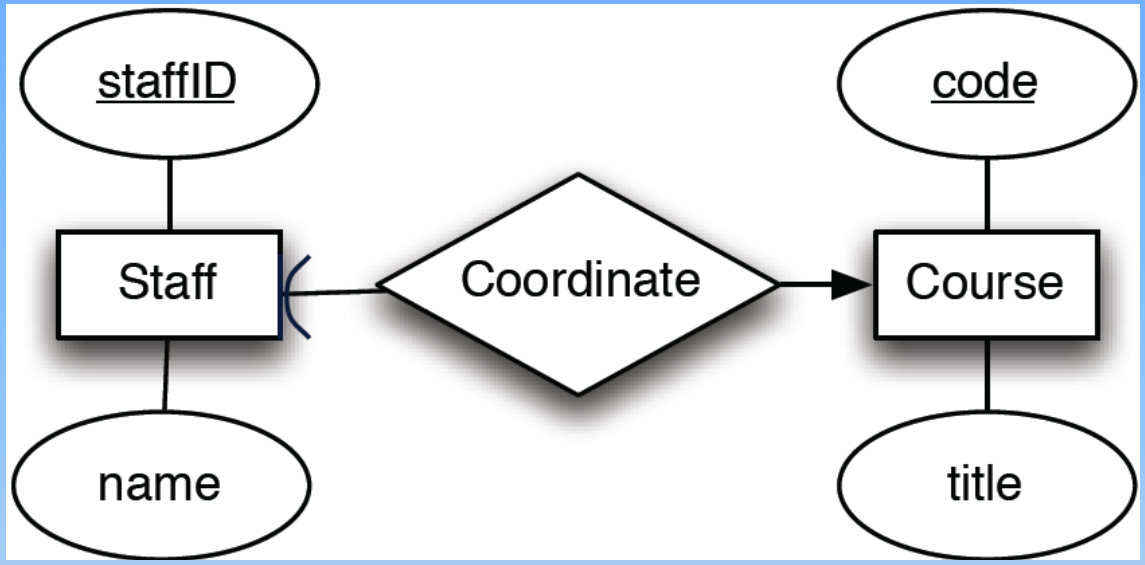


- Merging relationship relation with relation other than the "exact one" side.
  - No nulls.



- Merging relationship relation with either side.
  - No nulls.

Example



Staff(staffID, name)
Course(code, title)
Coordinate(staffId*, code*)

Course and Coordinate can be combined:

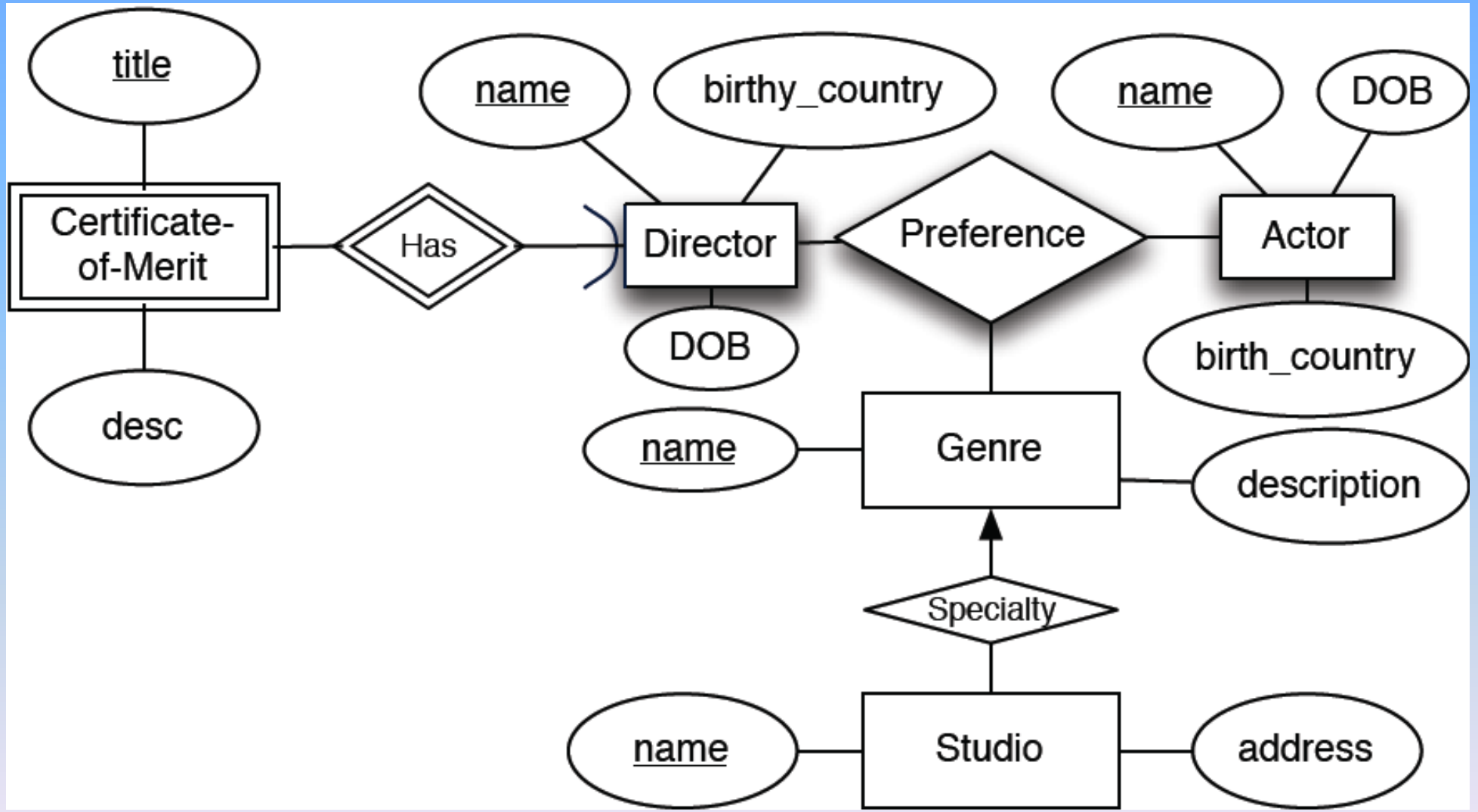
Staff(staffID, name)
Course(code, title, staffID*)

Summary

Relational database design practice:

1. Design an ER diagram.
2. Map the ER diagram into a relational database schema (following the mapping rules).
   - As most FDs are represented in ER models, if mapped correctly, the resulting relational model is most likely in BCNF or 3NF.
3. Examine the normal form of resulting relations, using FDs derived from the ER diagrams and FDs not in the ER diagram.
4. If a relation is not in BCNF/3NF, decompose it into BCNF/3NF.

A New Movie database: ER Diagram



A New Movie Database: Relational Database Schema

The 8 objects in the ER diagram are mapped to 6 relations:

Director(name, DOB, birth_country)
Actor(name, DOB, birth_country)
Genre(name, description)
Studio(name, address, specialtygenre*)
Certificate-of-Merit(directorName*, title, desc)
Preference (directorName*, actorName*, genreName*)

Acknowledgements

- Some slides are based on the lecture slides by J. D. Ullman.